Project no. 027087

TENCompetence

Building the European Network for Lifelong Competence Development

Project acronym: Integrated Project TENCompetence

Thematic Priority: 2.4.10

**D6.1 – ANNEX 1 - IMS LD Authoring**

Due date of deliverable: 31-05-2007
Actual submission date: 03-01-2008

Start date of project: 01-12-2005   Duration: 4 years

University of Bolton

Version 1.0

<table>
<thead>
<tr>
<th>Dissemination Level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PU</td>
<td>Public</td>
</tr>
<tr>
<td>PP</td>
<td>Restricted to other programme participants (including the Commission Services)</td>
</tr>
<tr>
<td>RE</td>
<td>Restricted to a group specified by the consortium (including the Commission Services)</td>
</tr>
<tr>
<td>CO</td>
<td>Confidential, only for members of the consortium (including the Commission Services)</td>
</tr>
</tbody>
</table>
D6.1 Annex 1 - IMS LD Authoring

Work package
WP6, Learning activities and units of learning

Task
Task 6.1, Task 6.2, Task 6.3 and Task 6.4

Date of delivery
Contractual: 31-05-2007 Actual: 03-01-2008

Code name
Version: 1.0 Draft [ ] Final [x]

Type of deliverable annex
Report

Security (distribution level)
Public

Contributors
David Griffiths, Tom Franklin, Paul Sharples, Phillip Beauvoir, Yongwu Miao, Colin Tattersall, Rob Koper, Krassen Stefanov, Milen Petrov, Adelina Aleksieva - Petrova, Toni Navarrete, Helena Batlle, Demetrios Sampson, Pythagoras Karampiperis, Judith Schoonenboom, Marco Luccini, Linda Napoletano

Authors (Partner)
UB, OUNL, SU, FBM-UPF, CERTH/ITI, UvA, ILABS

Contact Person
David Griffiths (UB)

WP/Task responsible
David Griffiths (UB)

EC Project Officer
Mr. Martin Májek

Abstract
This is Annex 1 to TENCompetence deliverable D6.1.

(for dissemination)
It consists of a detailed report on work carried out on the TENCompetence Authoring Component in the first 18 months of the project.

Keywords List
IMS Learning Design Authoring, Assessment Model, Run-time IMS LD Services, Pedagogical Templates, Units of Learning
## TABLE OF CONTENTS

1. SUMMARY OF THIS ANNEX .......................................................................................................................... 3

2. SUPPORTING EDUCATIONAL PRACTITIONERS IN DESIGNING FOR LIFELONG COMPETENCE DEVELOPMENT .................................................................................................................. 4

3. USE CASES FOR IMS LEARNING DESIGN AUTHORING ................................................................................. 6

   3.1 EXPRESSING AN EDUCATIONAL SCENARIO ............................................................................................... 7
   3.2 REPRESENTING AN EDUCATIONAL SCENARIO IN A MACHINE INTEROPERABLE FORMAT ....................... 11
   3.3 POPULATING AN EDUCATIONAL SCENARIO WITH EDUCATIONAL RESOURCES .................................. 14
   3.4 SHARING AN EDUCATIONAL SCENARIO AND/OR UNIT OF LEARNING WITHIN A COMMUNITY OF EDUCATIONAL PRACTICE ....................................................................................... 17

4. REVIEW OF EXISTING IMS LD AUTHORING TOOLS ...................................................................................... 19

   4.1 1ST GENERATION IMS LD AUTHORING TOOLS ......................................................................................... 19
   4.2 2ND GENERATION IMS LD AUTHORING TOOLS ......................................................................................... 19
   4.3 OVERVIEW OF IMS LD AUTHORING TOOLS ............................................................................................. 21

5. PROPOSED ARCHITECTURE FOR IMS LD AUTHORING TOOL ........................................................................... 24

   5.2 IMS LD AUTHORING CONCLUSIONS AND FUTURE WORK ......................................................................... 29

6. RECOURSE V.1.0, THE TENCOMPETENCE LEARNING DESIGN AUTHORING TOOL ...................................... 30

7. DESIGN REQUIREMENTS FOR THE WP6 LD EDITOR ...................................................................................... 34

   7.1 INTRODUCTION ............................................................................................................................................. 35
   7.2 PLACEMENT OF THE LD EDITOR .................................................................................................................. 37
   7.3 THE ORGANISER VIEW ................................................................................................................................. 37
   7.4 THE LD EDITOR WINDOW .............................................................................................................................. 38
   7.5 OVERVIEW EDITOR PAGE .............................................................................................................................. 39
   7.6 “COURSE” EDITOR PAGE .............................................................................................................................. 39
   7.7 ACTIVITIES EDITOR PAGE ............................................................................................................................. 42
   7.8 ENVIRONMENTS EDITOR PAGE .................................................................................................................... 45
   7.9 ROLE EDITOR PAGE ........................................................................................................................................ 45
   7.10 CODE EDITOR PAGE ...................................................................................................................................... 46
   7.11 FILES AND RESOURCES ORGANISER VIEW ............................................................................................... 46
   7.12 LD PREVIEW WINDOW ................................................................................................................................. 46
   7.13 ITEM EDITOR/VIEWER ................................................................................................................................. 47
   7.14 CODE ARCHITECTURE ................................................................................................................................. 49
   7.15 FILE FORMATS ............................................................................................................................................. 49
   7.16 SERIALISATION ........................................................................................................................................... 49
   7.17 DATABASE .................................................................................................................................................. 49
   7.18 TECHNICAL CONSIDERATIONS .................................................................................................................. 49

8. EVALUATION OF RECURSE, TENCOMPETENCE AUTHORING COMPONENT ............................................... 51

   8.1 INTRODUCTION ............................................................................................................................................. 51
   8.2 METHODOLOGY ............................................................................................................................................ 53
   8.3 QUESTIONNAIRE ANALYSIS .......................................................................................................................... 55
   8.4 FREE TEXT RESPONSES ................................................................................................................................. 59
   8.5 FURTHER ANALYSIS ...................................................................................................................................... 61
   8.6 EXPERT INSPECTION ..................................................................................................................................... 62
   8.7 CONCLUSIONS ............................................................................................................................................... 63
   8.8 APPENDIX 1 TO RECURSE EVALUATION: PARTICIPANT WORKSHEET ....................................................... 65
   8.9 APPENDIX 2: QUESTIONNAIRE ................................................................................................................... 84

9. THE CONTRIBUTION MADE BY THE RECURSE LD EDITOR ............................................................................. 86
1 Summary of this Annex

This annex provides documentation of the work carried out in the area of IMS Learning Design Authoring in the first 18 months of the TENCompetence project.

Sections 2, 3 and 4 describe the preparatory analysis which informs the work carried out. This includes presentation of use cases for IMS Learning Design (LD) authoring, and a review of existing IMS LD tools.

Section 5 details the architecture proposed for the LD Authoring tool in month 12, in which an inference engine which would interpret an activity workflow diagram would take a central role. As described in the main body of D 6.1, this proposed architecture had to be adapted, due to the non-availability of key personal at a project partner who could implement the inference engine.

Sections 6 to 9 describe work on the ReCourse LD Editor. Section 6 describes the revised strategy in this task, based around a redesign of the Reload LD Editor. In Section 7 the design requirements for the LD Editor are provided. Section 8 describes evaluation work carried out with authentic users using ReCourse. Finally, the contribution made by the ReCourse LD Editor is summarised.

The code for the ReCourse LD Editor, and the end user documentation, is available at http://www.tencompetence.org/ldauthor/
2 Supporting educational practitioners in designing for lifelong competence development

The term “pedagogical model for competence development” defines the context in which learning activities are carried out, rather than constraining or guiding the choice of pedagogies to be used in any given context (David Griffiths, TENCompetence Initial Review of Pedagogic Models, pp. 6). In TENCompetence, there is a consensus that there is no scientific basis for providing such guidance which is valid for all circumstances, and so it is more appropriate to provide design methods and tools which support learning designers and practitioners in their practice (Rob Koper, WP6 T6.1 Forum Posts, 13 and 14 April 2006). Thus, flexibility in IMS LD Authoring in the context of lifelong competence development is defined as the provision of an inclusive set of methods/tools for learning designers and practitioners in order to support them in using their pedagogical skills to develop learning activities which assist in attaining the desired competences.

In this context, the goal of TENCompetence WP6 IMS LD Authoring Working Group is to provide a framework for supporting learning designers and practitioners through the entire lifecycle of the authoring process, that is, the process of expressing educational scenarios using commonly recognized terms, representing these educational scenarios using a common and interoperable format, and sharing them within a Community of Educational Practice. To achieve this goal, the following key actions have been identified:

- **Definition of Methodological Steps for Expressing Educational Scenarios for Lifelong Competence Development.** Effective sharing of Educational Practices in a Community of Training Practitioners has not yet been realized (Griffiths, 2005), due to the fact that a common and interoperable expression framework for describing such practices does not exist. As a consequence, although practitioners have the ability to share Learning Activities and Units of Learning (via the use of the IMS Learning Design conformant infrastructure), they face difficulties in interpreting the underlying educational approach(es) used, and they cannot reuse them neither when delivering activities to their end-users, nor when designing similar learning activities for a different context of use.

- **Design and Development of a High Level Graphical IMS LD Authoring Tool.** Despite the wide adoption of the IMS Learning Design specification, and the existence of a number of IMS LD based Authoring Tools (such as Reload LD Editor, CopperAuthor, ASK-LDT, MOT+), still practitioners face difficulties when designing Units of Learning (Griffiths, 2005). This is due to the fact that practitioners have difficulty in understanding the language of the IMS Learning Design specification that is needed in order to create Units of Learning. Thus, higher-level graphical Learning Flow tools are needed, which enable practitioners visualize and assemble UOLs more easily (Bill Olivier, WP6 White paper, pp. 9-11). The design paradigm of these tools should be close to common practices on designing pedagogical scenarios, rather than to the XML-based structure of the IMS Learning Design specification.

To this end, we start from the definition of exemplary design steps for expressing the main elements of an Educational Scenario when designing Units of Learning for lifelong competence development, and we present examples of vocabularies of common terms that can be used to support the expression process. After that, we apply these design steps in the case of two well known educational strategies (namely, problem-based learning and project-
based learning), towards the definition of exemplary Pedagogical Templates that can be used for the case of lifelong competence development. Moreover, we represent these Pedagogical Templates in an interoperable format, that is, the IMS Learning Design specification, integrating the presented Vocabularies of Common Terms with the Information Model of the IMS LD specification.

Finally, based on the presented methodological steps for designing Learning Activities and/or Units of Learning for lifelong competence development, we provide Use Cases of the envisioned IMS LD Authoring Tool, we compare the existing IMS LD implementations based on how well they address the identified Use Cases, and present the proposed architectural design of the TENCompetence IMS LD Authoring Tool, towards addressing the authoring needs for learning designers and educational practitioners expressed in the Use Cases.
3 Use Cases for IMS Learning Design Authoring

From the discussion of pedagogic models in the previous section, and the methodological steps identified there for designing Educational Scenarios and/or Units of Learning for lifelong competence development, it is evident that the lifecycle of the authoring process consists of the following key phases:

- *Express an Educational Scenario* with common terms, that can be identified by learning designers and educational practitioners
- *Represent an Educational Scenario in a machine interoperable format*, that is, the IMS Learning Design specification
- *Populate an Educational Scenario with Educational Resources*, so as to create a complete Unit of Learning
- *Share an Educational Scenario and/or Unit of Learning* within a Community of Educational Practice.

In this section, we present exemplary Use Cases of an envisioned IMS LD Authoring Tool, aiming to support all the above mentioned phases of the authoring process. These Use Cases are presented in a tabular form, providing a narrative description and the flow of events for each one of them.
3.1 Expressing an Educational Scenario

- Expert Learning Designer
  - Create an Educational Scenario from Scratch
  - Express an Educational Scenario
    - <<include>>
    - Modify an existing Educational Scenario
      - <<include>>
      - <<include>>
      - Reuse existing Educational Scenario(s) for creating more complex ones
- Educational Practitioner
- Individual Learner
### 1. Create an Educational Scenario from Scratch

<table>
<thead>
<tr>
<th>Description</th>
<th>A user of the IMS LD Authoring Tool wants to create an Educational Scenario from scratch.</th>
</tr>
</thead>
</table>
| Exemplary Actors | *Formal Learning:* Expert Learning Designer, Educational Practitioner  
*Informal Learning:* Individual Learner |
| Basic Flow | The user initializes the IMS LD Authoring Tool and selects the “New Scenario” option from the menu.  
The user defines an Educational Scenario by following the next steps:  
1. Specify the Learning Objectives of this scenario using a Taxonomy of Learning Objectives  
2. Specify the participating Roles in this scenario  
3. Specify Environments (tools and services) used in this scenario  
4. Specify Activities of this Scenario and describe them using a Taxonomy of Learning Activities  
5. Specify the flow of Activities using a graphical representation of the learning flow  
6. Specify Conditions upon the flow of activities based on defined properties  
7. The user saves the Educational Scenario as a basis for the production of Units of Learning. |
| Alternative Flow | The user saves the Educational Scenario as a component (into a pool of available components) to be used for the definition of more complex scenarios (see Use Case 2). |
| Pre-Conditions | The user should be aware of the Taxonomies and the Graphical Notation used. |
| Post-Conditions | None |
| Specific Requirements | Common Taxonomy of Learning Objectives  
Common Taxonomy of Learning Activities  
Graphical Notation for representing the Learning Flow |
| Include | None |
## 2. Reuse existing Educational Scenario(s) for creating more complex ones

<table>
<thead>
<tr>
<th>Description</th>
<th>A user of the IMS LD Authoring Tool wants to create an Educational Scenario, by reusing pre-existing Educational Scenario(s) as components of the new scenario.</th>
</tr>
</thead>
</table>
| Exemplary Actors | **Formal Learning**: Expert Learning Designer, Educational Practitioner  
**Informal Learning**: Individual Learner |
| Basic Flow | The user initializes the IMS LD Authoring Tool and selects the “New Scenario” option from the menu.  
The user defines an Educational Scenario by following the next steps:  
1. Specify the Learning Objectives of this scenario using a Taxonomy of Learning Objectives  
2. Specify the participating Roles in this scenario  
3. Specify Environments (tools and services) used in this scenario  
4. Select from a pool of available components (pre-existing Educational Scenarios) the ones that he/she wants to reuse, by interpreting them via the Taxonomy of Learning Activities used.  
5. Specify new Activities for this Scenario and describe them using a Taxonomy of Learning Activities  
6. Specify the flow of Components/Activities using a graphical representation of the learning flow  
7. Specify Conditions upon the learning flow based on properties defined  
The user saves the Educational Scenario as a basis for the production of Units of Learning. |
| Alternative Flow | The user saves the Educational Scenario as a component (into a pool of available components) to be used for the definition of another more complex scenario. |
| Pre-Conditions | The user should be aware of the Taxonomies and the Graphical Notation used. |
| Post-Conditions | None |
| Specific Requirements | 1. Common Taxonomy of Learning Objectives  
2. Common Taxonomy of Learning Activities  
3. Graphical Notation for representing the Learning Flow |
| Include | None |
### 3. Modify an existing Educational Scenario

<table>
<thead>
<tr>
<th>Description</th>
<th>A user of the IMS LD Authoring Tool wants to create an Educational Scenario, by modifying an existing one.</th>
</tr>
</thead>
</table>
| Exemplary Actors | *Formal Learning*: Expert Learning Designer, Educational Practitioner  
*Informal Learning*: Individual Learner |
| Basic Flow | The user initializes the IMS LD Authoring Tool and selects the “Open Scenario” option from the menu.  
The user modifies an Educational Scenario by following the next steps:  
1. Select from a pool of available components (pre-existing Educational Scenarios) the one that he/she wants to modify, by interpreting it via the Taxonomy of Learning Activities used.  
2. Specify new Activities in this Scenario (or modify existing ones) and describe them using a Taxonomy of Learning Activities  
3. Modify (if needed) the flow of Activities using a graphical representation of the learning flow  
4. Modify (if needed) Conditions upon the learning flow based on properties defined  
The user saves the Educational Scenario as a basis for the production of Units of Learning. |
| Alternative Flow | The user saves the Educational Scenario as a component (into a pool of available components) to be used for the definition of more complex scenarios (see Use Case #2). |
| Pre-Conditions | The user should be aware of the Taxonomies and the Graphical Notation used. |
| Post-Conditions | None |
| Specific Requirements | 1. Common Taxonomy of Learning Objectives  
2. Common Taxonomy of Learning Activities  
3. Graphical Notation for representing the Learning Flow |
| Include | None |
3.2 Representing an Educational Scenario in a Machine Interoperable Format

- Expert Learning Designer
- Educational Practitioner
- Individual Learner

- Exporting an Educational Scenario to IMS LD
- Importing an IMS LD Educational Scenario
4. Exporting an Educational Scenario to IMS LD

<table>
<thead>
<tr>
<th>Description</th>
<th>A user of the IMS LD Authoring Tool wants to export an Educational Scenario in IMS Learning Design, so that he/she can (a) use any low level IMS LD Authoring Tool to modify/ refine it, (b) use any IMS LD compliant run-time engine to illustrate its use and/or (c) upload it to any IMS LD compliant Web Repository of Learning Designs, so as to share it with others.</th>
</tr>
</thead>
</table>
| Exemplary Actors | *Formal Learning*: Expert Learning Designer, Educational Practitioner  
*Informal Learning*: Individual Learner |
| Basic Flow | 1. The user initializes the IMS LD Authoring Tool and selects the “Export to LD” option from the menu.  
2. The user exports the desired Educational Scenario by following the next steps:  
3. [Optional] Create the desired Educational Scenario (see Uses Cases 1, 2 and 3)  
4. Select the local path for the desired IMS LD manifest  
5. Save the desired Educational Scenario in IMS Learning Design XML language |
| Alternative Flow | None |
| Pre-Conditions | None |
| Post-Conditions | None |
| Specific Requirements | None |
| Include | Use Case 1: Create an Educational Scenario from Scratch  
Use Case 2: Reuse existing Educational Scenario(s) for creating more complex ones  
Use Case 3: Modify an existing Educational Scenario |
## 5. Importing an IMS LD Educational Scenario

<table>
<thead>
<tr>
<th>Description</th>
<th>A user of the IMS LD Authoring Tool wants to import an IMS LD Educational Scenario so as to be used (a) “as is” for the production of Units of Learning, (b) as a component for the definition of other more complex Educational Scenarios.</th>
</tr>
</thead>
</table>
| Exemplary Actors | **Formal Learning**: Expert Learning Designer, Educational Practitioner  
**Informal Learning**: Individual Learner |
| Basic Flow | The user initializes the IMS LD Authoring Tool and selects the “Import LD” option from the menu.  
The user imports the desired Educational Scenario (represented in IMS LD) by following the next steps:  
1. Select the desired IMS LD Educational Scenario (xml manifest)  
2. Interpret the Learning Objectives of this scenario via the elements of the Taxonomy of Learning Objectives used  
3. Visualize the Learning Flow described within the selected Educational Scenario, to attain an overview of the defined learning process  
4. Interpret the semantics of the activities contained in the Educational Scenario via the elements of the Taxonomy of Learning Activities used.  
The user saves the Educational Scenario as a basis for the production of Units of Learning. |
| Alternative Flow | The user saves the Educational Scenario as a component (into a pool of available components) to be used for the definition of more complex scenarios (see Use Case #2). |
| Pre-Conditions | The user should be aware of the Taxonomies and the Graphical Notation used. |
| Post-Conditions | None |
| Specific Requirements | 1. Common Taxonomy of Learning Objectives  
2. Common Taxonomy of Learning Activities  
3. Graphical Notation for representing the Learning Flow |
| Include | None |
3.3 Populating an Educational Scenario with Educational Resources

- Expert Learning Designer
- Educational Practitioner
- Individual Learner

Populate an Educational Scenario with Educational Resources

Generate Units of Learning

<<include>>

Modify the Educational Resources in an existing UoL

<<include>>
## 6. Generate Units of Learning

<table>
<thead>
<tr>
<th>Description</th>
<th>A user of the IMS LD Authoring Tool creates an Educational Scenario, or selects an existing one from a list of available scenarios, and populates it with educational resources in order to generate a complete Unit of Learning.</th>
</tr>
</thead>
</table>
| Exemplary Actors                                 | Expert Learning Designer, Educational Practitioner  
Informal Learning: Individual Learner |
| Basic Flow                                       | The user initializes the IMS LD Authoring Tool and selects the “Generate UOL” option from the menu.  
The user generates a Unit of Learning by following the next steps:  
1. [Optional] Create the desired Educational Scenario (see Uses Cases 1, 2 and 3)  
2. Select from a pool of available Educational Scenarios the desired one, by interpreting it via the Taxonomy of Learning Activities used.  
3. Select an Activity within the Educational Scenario  
4. Interpret the semantics of the selected activity via the elements of the Taxonomy of Learning Activities used.  
5. Inspect the activity metadata to find the type of the required resources for the selected activity  
6. Assign appropriate resources to the selected activity  
7. Repeat steps 3 through 6 for all activities of the Educational Scenario  
8. Select the local path and filename to save the desired Unit of Learning  
9. Save the Unit of Learning as an IMS Learning Design Content Package (zip format) |
| Alternative Flow                                 | None |
| Pre-Conditions                                   | The user should be aware of the Taxonomies used. |
| Post-Conditions                                  | None |
| Specific Requirements                            | Common Taxonomy of Learning Activities |
| Include                                          | Use Case 1: Create an Educational Scenario from Scratch  
Use Case 2: Reuse existing Educational Scenario(s) for creating more complex ones  
Use Case 3: Modify an existing Educational Scenario |
| **7. Modify the Educational Resources in an existing Unit of Learning** |
|-----------------|--------------------------------------------------------------------------------------------------|
| **Description** | A user of the IMS LD Authoring Tool wants to modify the educational resources used in an existing Unit of Learning. |
| **Exemplary Actors** | Formal Learning: Expert Learning Designer, Educational Practitioner  
Informal Learning: Individual Learner |
| **Basic Flow** | The user initializes the IMS LD Authoring Tool and selects the “Open UOL” option from the menu.  
The user modifies an existing Unit of Learning by following the next steps:  
1. Select the desired Unit of Learning  
2. Select the Activity(ies) for which the educational resources need to be modified/ updated within the selected UOL  
3. Interpret the semantics of the selected activity(ies) via the elements of the Taxonomy of Learning Activities used.  
4. Inspect the activity metadata to find the type of the required resources for the selected activity (ies)  
5. Assign new resources to the selected activity  
6. Select the local path and filename to save the desired Unit of Learning  
7. Save the Unit of Learning as an IMS Learning Design Content Package (zip format) |
| **Alternative Flow** | None |
| **Pre-Conditions** | The user should be aware of the Taxonomies used. |
| **Post-Conditions** | None |
| **Specific Requirements** | Common Taxonomy of Learning Activities |
| **Include** | None |
3.4 Sharing an Educational Scenario and/or Unit of Learning within a Community of Educational Practice

- Expert Learning Designer
- Educational Practitioner
- Individual Learner

<<include>>

Publish an Educational Scenario and/or UoL in a Web Repository
### 8. Publish an Educational Scenario and/or UOL in a Web Repository

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>A user of the IMS LD Authoring Tool wants to share an Educational Scenario and/or Unit of Learning within a Community of Educational Practice.</th>
</tr>
</thead>
</table>
| **Exemplary Actors** | Formal Learning: Expert Learning Designer, Educational Practitioner  
Informal Learning: Individual Learner |
| **Basic Flow** | The user initializes the IMS LD Authoring Tool and selects the “Publish” option from the menu.  
The user publishes an Educational Scenario and/or Unit of Learning to a Web Repository of learning designs by following the next steps:  
1. Select the desired Educational Scenario and/or Unit of Learning to be shared within a Community of Educational Practice  
2. Upload the Educational Scenario and/or Unit of Learning to an IMS LD compliant Web Repository |
| **Alternative Flow** | None |
| **Pre-Conditions** | None |
| **Post-Conditions** | None |
| **Specific Requirements** | None |
| **Include** | None |
4 Review of existing IMS LD Authoring tools

In this section we present an overview of the existing IMS LD Authoring Tools, aiming to identify how well they cover the Use Cases presented in previous section. The currently available IMS LD Authoring Tools could be classified into two main classes based on the authoring philosophy they implement, namely, first generation and second generation IMS LD authoring tools.

4.1 1st Generation IMS LD Authoring Tools

The fist generation of IMS LD Authoring Tools, includes tools which provide form-based interfaces for the definition of Educational Scenarios and/or Units of Learning, using the XML structure of the IMS Learning Design specification as the main driver of the authoring process. Depending on the implementation, these tools are tab-structured or tree-based. The main advantage of these tools is that they provide direct control of the Learning Design information model elements. However, they are rather difficult to be used by less experienced designers and they require pre-processing of the structure of the desired scenario in order for a designer to be able to express it directly in XML notation.

Examples of First Generation IMS LD Authoring Tools include:

- The Reload project (http://www.reload.ac.uk/) is built on the developments of the Valkenburg Group (http://www.valkenburggroup.org) concerning Educational Modelling Language (EML) and the associated IMS Learning Design Specification. The Reload Editor, supports authoring of all IMS Learning Design levels through form-based user interfaces (Olivier, B., 2004). Additionally, it integrates a content packaging mechanism that allows exporting of a learning design in the form of a content package.

- The Alfanet project (http://alfanet.ia.uned.es) has developed an authoring tool supporting IMS Learning Design level A and B, which allows a designer to create adaptive Units of Learning. The Alfanet LD Editor is designed reflecting the structure of the IMS LD information model. As a result it provides flexibility in defining activity properties and adaptivity conditions, in balance of user-friendliness (Griffiths, D., Blat, J., Garcia, R., Vogten, H., Wong, K.L., 2005).

- CopperAuthor (http://sourceforge.net/projects/copperauthor) is an open-source form-based editor developed by the Open University of The Netherlands. CopperAuthor supports IMS Learning Design Level A and B, and calls CopperCore (a runtime IMS LD engine developed in the context of the Alfanet project) through web-services to support validation and play of the created Units of Learning (Olivier, B., 2004).

4.2 2nd Generation IMS LD Authoring Tools

The second generation of IMS LD Authoring Tools, includes tools which provide graphical-based, drag-and-drop interfaces for the definition of Educational Scenarios and/or Units of Learning. Their main advantage is that they support the design process without requiring pre-existing knowledge of the details of the IMS Learning Design information model. On the other hand, the non-so-close to the XML structure of the IMS Learning Design specification character of these tools, restrains them from offering the full functionality of the IMS LD modelling language to users that need it. Additionally, due to the implementation limitations imposed by the complexity of IMS LD levels higher than level A, most of these tools are
only capable of representing Educational Scenarios and/or Units of Learning conformant to IMS LD Level A. In addition, they are able of performing one way conversions, that is, they generate the IMS LD manifest from a graphical representation of the learning flow but not the other way around; these tools are not capable of carrying out the transformation of the IMS manifest to the corresponding graphical representation.

Examples of Second Generation IMS LD Authoring Tools include:

- The MOT+ editor (http://www.unfold-project.net/general_resources_folder/tools/mot) has been developed by the University of Quebec and provides a graphical user interface for describing instructional scenarios. The MOT+ editor can be configured to create learning designs conforming to IMS LD level A (Paquette, G., Teja, I., Leonard, M., Lundgren-Cayrol, K., Marino, O., 2005).

- The LAMS (Learning Activity Management System - www.lamsinternational.com) is a learning management system inspired by EML and IMS Learning Design that provides an authoring tool based on the use of drag-and-drop user interfaces (Dalziel, J.R., 2003, Olivier, B., 2004). The limitation of this environment is that it supports only linear sequence of pre-defined activities, thus cannot implement complex IMS LD level B Educational Scenarios and/or Units of Learning. However, an export feature in IMS LD Level A has been implemented (UNFOLD Report, 2005).

- The ASK Learning Designer Toolkit (ASK-LDT) (Karampiperis, P., Sampson, D., 2004, Sampson, D., Karampiperis, P., Zervas, P., 2005) is an authoring tool supporting the use of the IMS LD Level A and B specification as the other tools. Although graphical based, it is not as far from the LD specification. This is because its design philosophy allows developers to use a high-level notation in order to define learning scenarios, whereas it internally uses a low level XML-based notation language to describe IMS LD elements. Thus, this tool may be considered as an implementation lying on both worlds: First and Second Generation LD authoring tools. However, this tool does not avoid the pitfalls of either the First Generation of IMS LD authoring tools, since its expression dynamics revolve around the IMS LD specification, or the Second Generation of IMS LD authoring tools, since it does not offer the full functionality of the IMS LD modelling language.
4.3 Overview of IMS LD Authoring Tools

The overview classification of existing IMS LD Authoring Tools is presented in the table below. From this figure, we can observe that the First Generation of IMS LD Authoring Tools (e.g. Reload LD Editor, Alphanet Editor and CopperAuthor) are closer to the IMS Learning Design specification than the Second Generation of IMS LD Authoring Tools (e.g. MOT+, LAMS and ASK-LDT).
Comparison of existing IMS LD Authoring Tools

The following matrix presents the list of Use Cases presented in section 4 and their coverage by the existing IMS LD Authoring Tool implementations.

<table>
<thead>
<tr>
<th>Use Case per Authoring Tool</th>
<th>1stGeneration Tools</th>
<th>2ndGeneration Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reload</td>
<td>Alfamet</td>
</tr>
</tbody>
</table>

### Expressing an Educational Scenario

1. Create an Educational Scenario from Scratch

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+/?</td>
<td>+/?</td>
<td>+/?</td>
</tr>
</tbody>
</table>

2. Reuse existing Educational Scenario(s) for creating more complex ones

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+/?</td>
</tr>
</tbody>
</table>

3. Modify an existing Educational Scenario

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+/?</td>
</tr>
</tbody>
</table>

### Representing an Educational Scenario in a Machine Interoperable Format

4. Exporting an Educational Scenario to IMS LD

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>+</td>
<td>+</td>
<td>√/?</td>
</tr>
</tbody>
</table>

5. Importing an IMS LD Educational Scenario

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>+/?</td>
</tr>
</tbody>
</table>

### Populating an Educational Scenario with Educational Resources

6. Generate Units of Learning

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√?</td>
<td>√</td>
</tr>
</tbody>
</table>

7. Modify the Educational Resources in an existing Unit of Learning

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>√</td>
</tr>
</tbody>
</table>

### Sharing an Educational Scenario and/or Unit of Learning within a Community of Educational Practice

8. Publish an Educational Scenario and/or UOL in a Web Repository

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Reload</th>
<th>Alfamet</th>
<th>F-Autho</th>
<th>MOT</th>
<th>LAM</th>
<th>ASK-LDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.</td>
<td>-</td>
<td>-</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1: IMS LD Authoring Use Case Coverage Matrix

<table>
<thead>
<tr>
<th>Legend</th>
</tr>
</thead>
<tbody>
<tr>
<td>√</td>
</tr>
<tr>
<td>+</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>?</td>
</tr>
</tbody>
</table>
Within TENCompetence we require second generation functionality, but none of the present tools is fully satisfactory: Mot+ is a complex tool for experts, LAMS is not fully compliant with IMS LD (although progress is being made in this direction), and ASK LDT is a prototype which has not yet been released under an Open Source license. Consequently there is a clear need for the project to develop its own authoring solution.
5 Proposed Architecture for IMS LD Authoring Tool

From the comparison of the existing IMS LD Authoring Tools presented in the previous section, it is evident that an authoring tool addressing the requirements of the entire lifecycle of the authoring process does not exist. To this end, in WP6 Working Group A (IMS LD Authoring) we aim to design and develop a new IMS LD Authoring Tool that will support learning designers and practitioners through the entire lifecycle of the authoring process, that is, the process of expressing educational scenarios using commonly recognized terms, representing these educational scenarios using a common and interoperable format, and sharing them within a Community of Educational Practice.

In this section we present the architectural design of the envisioned IMS LD Authoring Tool (referred to as TenC IMS LDAT), as it was conceived in M 6.1, in month 12 of the project.

5.1.1 Overview Description of Proposed Architecture

As described in previous sections, the lifecycle of the authoring process consists of the following key phases:

- Express an Educational Scenario with common terms, that can be identified by learning designers and educational practitioners
- Represent an Educational Scenario in a machine interoperable format, that is, the IMS Learning Design specification
- Populate an Educational Scenario with Educational Resources, so as to create a complete Unit of Learning
- Share an Educational Scenario and/or Unit of Learning within a Community of Educational Practice.

The proposed architecture of the TENCompetence IMS LDAT has been defined in a modular way. Such an approach enables the easier extension of the implemented functionalities, as well as, the reuse of individual components/modules from other open source implementations. The scope of each component/module defined is to directly address the requirements of the relevant authoring process phase.

As a result, the architectural design of the TENCompetence IMS LDAT consists of the following main components/modules:

1. A Graphical User Interface module that allows the user to represent graphically the learning flow of an Educational Scenario, as well as, express the defined activities with common terms.
2. An Inference Engine that interprets the activity workflow defined in an Educational Scenario and transforms it to the IMS Learning Design XML language and vice-versa.
3. A Packaging Engine that enables the association of activities specified in an Educational Scenario with educational resources, and generates Units of Learning (IMS LD compliant content packages in zip format).
4. A Publishing Engine that is responsible for making available the created Educational Scenarios and/or Units of Learning on a Web Repository.
5. IMS LD Authoring Process Phase | Responsible TENCompetence IMS LDAT module
---|---
A: Express an Educational Scenario | Graphical User Interface module
B: Represent an Educational Scenario in a machine interoperable format | Inference Engine
C: Populate an Educational Scenario with Educational Resources | Packaging Engine
D: Share an Educational Scenario and/or Unit of Learning | Publishing Engine

Table 2: Mapping of TENCompetence IMS LDAT components/modules to IMS LD Authoring Process Phases

In the figure that follows the interconnection between the TENCompetence IMS LDAT modules is presented.

![Diagram of TENCompetence IMS LD Authoring Tool Architecture](image)

**Figure 2: The TenC IMS LD Authoring Tool Architecture**

### 5.1.2 Detailed description of individual modules

#### 5.1.2.1 The Graphical User Interface module

The Graphical User Interface module is essential for providing the user with an accessible means of interaction with the TENCompetence IMS LD Authoring tool. The design principles of this GUI are (a) simplicity of the user interface and the representation of a learning flow, so as to enable learning designers and educational practitioners have a better understanding of the designed learning process, (b) hide IMS LD elements that can be directly inferred from the high level design from the user, so as to enable easier assembly of Educational Scenarios and Units of Learning.
The components that comprise the GUI module are the graphical editors for the Activities, Roles and Environments, and the ones that are responsible for the serialization. Accordingly, the functionalities of this module are grouped in the same manner. For Activities, the GUI provides a Graphical Editor with the following basic set of functionalities:

- Definition of Activities and description according to a Taxonomy of Learning Activities
- Definition of the flow of Activities
- Definition of Activity properties and conditions upon the flow of activities
- Definition of a list of Roles participating in each Activity
- Definition of a list of Environments utilized for each Activity

The graphical editor for the Activities provides shapes that represent the Activities in the Activity design area. Activity shapes may be created by making use of the tools available in the tools palette of the editor. The created shapes are resizable and may be connected to each other by specifying types of connections, that is, normal connections and connections that are triggered by conditions. These connections specify the flow of Activities. Every time one of the Activities defined is selected, the Activity properties dialog box is updated to display the properties of the specific Activity. Many of these properties are editable (either through the dialog and/or through the shape representing the activity itself). Through the same dialog box, other dialog boxes are invoked to define the Roles participating in this Activity as well as the Environments for it.

For Roles, the GUI provides a Graphical Editor with the following basic set of functionalities:

- Definition of Roles
- Definition of the connections between Roles (expressing subtypes of roles)
- Definition of Role properties

The Roles Graphical Editor has more or less the same functionality as the editor for Activities. For creating Roles and relations between them, the editor provides a tools palette. This palette helps create Learner and Support Roles, as well as connections between them. Roles can be defined by dragging and dropping Role shapes within the design area of the editor. Once again, these shapes can be manipulated to change their size and properties such as the name of the Role. Only Roles of the same type, that is Support or Learner Roles, could be connected to one another to create sub-type relationships, in order to allow actors to play different roles in certain types of Activity. Properties for Roles may be edited through the properties dialog that is updated with specific Role properties when that Role (represented by its shape) is selected.

For Environments, the GUI provides a Graphical Editor with the following basic set of functionalities:

- Definition of Environments
- Definition of Learning Objects contained within an Environment

The graphical editor for Environments is more simplistic compared to the previous two editors. It too provides a Tools palette to create environment shapes. The user may create Environment shapes using the drag ‘n ‘drop functionality. The user may also provide a name for each environment by editing the environment shape within the design area of the editor.

As for the serialization component of the GUI module the functionalities are the following:

- Creating an Educational Scenario
- Exporting an Educational Scenario in IMS LD (through the Inference Engine module)
- Importing an Educational Scenario from IMS LD (through the Inference Engine module)
• Saving an Educational Scenario
• Loading an Educational Scenario (that has been created within the TenC IMS LD AT GUI)

Educational Scenarios created from the TENCompetence IMS LD Authoring Tool could be saved following user’s request. Saved Educational Scenarios may be loaded at any time using the load functionality of the module. The GUI also will support the import and export to IMS LD format. Importing an IMS LD representation of an Educational Scenario provides the user with its visual depiction by the GUI module. Exporting an Educational Scenario to IMS LD provides the means for making that entity “playable” by any IMS LD compatible run-time engine. The ability to save to IMS LD is possible through the Inference Engine module. In the same manner, loading a graphical representation of an IMS LD saved Educational Scenario is possible by providing the GUI module with the graphical representation of this Educational Scenario, as output from the Inference Engine module that reads in the relevant IMS LD manifest file.

5.1.2.2 The Inference Engine Module

The Inference Engine is responsible for inferring information from the GUI output, in order to generate the IMS LD XML manifest; it is also responsible for inferring GUI representation information from reading in an external IMS LD XML manifest. The process of inferring IMD LD elements (e.g. Acts, Roleparts and Activity Structures) from the higher level GUI representation of Roles and Activities, and vice-versa is non-trivial. For this reason the Inference Engine comprises a module of its own.

One of the main reasons for using an Inference Engine is keeping the GUI of the LD authoring tool user-friendly. This is done by hiding IMS-LD specific Information from the LD authoring tool user. The Inference Engine is capable of extracting IMS LD elements (e.g. Acts, Roleparts and Activity Structures) from a higher level workflow graph representing the flow of activities. The absence of those IMS LD elements from the interface that is presented to the user, makes the tool more user-friendly and easier to be used from users with less knowledge/experience of the XML structure of the IMS Learning Design specification. Apart from that, the Inference Engine provides a solution for rendering a graphical representation of an Educational Scenario by transforming the IMS LD manifest from an external source to the GUI high level representation.

The functionalities the Inference Engine module implements are connected to exporting an Educational Scenario into an IMS LD manifest and vice versa. While exporting to IMS LD, the Inference Engine is responsible for performing the following set of tasks:
• Parsing the high level representation generated by the GUI
• Breaking up the Play into Acts
• Defining Roleparts for each Act
• Defining Activity structures for each Rolepart of each Act
• Generating the IMS-LD XML manifest

For the first functionality, the Inference Engine uses a parser in order to read in the Activity and Role specific data present in the intermediate GUI output. For the second one, heuristic rules are applied to make clear which Activities are contained within which Acts. Having the Acts of the Play in place, the Inference Engine isolates each Role within each Act and defines Roleparts via the execution of a transformation algorithm. Then for each Rolepart, the Inference Engine defines Activity Structures. These structures contain more than one Activities executed by that Rolepart in a selective or sequential manner, thus forming
Sequence and/or Selection Activity Structures. Finally, with all the above information readily available, the Inference Engine constructs the IMS-LD XML manifest.

5.1.2.3 The Packaging Engine Module

The Packaging Engine is responsible for associating activities defined in an IMS-LD manifest with educational resources and producing IMS LD conformant Content Packages that can be executed in any IMS LD conformant run-time engine. This module is broken down into two components: the Resource Specification Interface component and the Resource Packaging component. The first is related to the process of associating educational resources to activities within an IMS LD manifest in order to create a Unit of Learning, and the second to the packaging of the manifest file and associated resources.

The list of functionalities for the Packaging Engine Module is grouped into two categories, following the separation between the Resource Specification Interface and the Resource Packaging components of the module. Through the Resource Specification Interface component the user specifies which resources are associated with activities of an Educational Scenario in order to create a Unit of Learning. This component provides the following functionalities:

- Drag & Drop functionality for associating resources
- Support of multiple files for each resource element
- View Resource Lists for Activities contained in an Educational Scenario, as well as, Learning Objects utilized within an Environment.

The first functionality enables the user to associate resources to an Activity of an Educational Scenario easily. In addition to that, the Resource Specification Interface allows for multiple files to be associated with each resource element, as for example, an html page that contains multiple images, videos, etc. Finally, since Activities and Environments may have resources associated with them, lists of the associated resources may be presented through a dialog for each Activity or Environment to the user. It is important to note that the Resource Specification Interface is integrated within the principal GUI module (although it is an individual element tied functionally only to the Packaging Engine module) of the TENCompetence IMS LD Authoring Tool. The Resource Packaging component has only one functionality:

- Generation of the IMS LD Content Packages (in zip format)

This allows the bundle of the IMS LD manifest file (defining the Educational Scenario) and its associated resources to be compressed within a single zip file. This content package may be stored on the user’s local file system for later use, or be given as input to the Publishing Engine module to be uploaded in a Web Repository of Learning Designs.

5.1.2.4 The Publishing Engine Module

The Publishing Engine is responsible for uploading the produced Educational Scenarios and/or Units of Learning to a Web Repository of Learning Designs. The Publishing Engine module enables the user to share Educational Scenarios and/or Units of Learning within a Community of Educational Practice. The Publishing Engine module provides the following basic set of functionalities:

- Publishing an Educational Scenario to a Web Repository of Educational Scenarios
- Publishing a Unit of Learning to a Web Repository of UOLs

The term “publishing” means uploading of Educational Scenarios and/or Units of Learning. Through the GUI component of this module the user accesses the functionalities offered by
the Publishing Engine module. Thus, the user is allowed to publish an IMS LD manifest (Educational Scenario) and/or a full IMS LD content package (Unit of Learning).

5.2 IMS LD Authoring Conclusions and Future Work

In TENCompetence there is a consensus that there is no scientific basis for providing guidance on the choice of pedagogies to be used in a given context, which is valid for all circumstances, and so it is more appropriate to provide design methods and tools which support learning designers and practitioners in their practice. Thus, flexibility in IMS LD Authoring in the context of lifelong competence development is defined as the provision of an inclusive set of methods/tools for learning designers and practitioners in order to support them in using their pedagogical skills to develop learning activities which assist in attaining the desired competences.

In this context, the goal for IMS LD authoring in TENCompetence is to provide a framework for supporting learning designers and practitioners through the entire lifecycle of the authoring process, that is, the process of expressing educational scenarios using commonly recognized terms, representing these educational scenarios using a common and interoperable format, and sharing them within a Community of Educational Practice.

Towards achieving this goal, this document defines a set of Methodological Steps for Expressing Educational Scenarios for Lifelong Competence Development, providing examples of vocabularies of common terms that can be used to support the expression process. Based on the presented methodological steps, this document identifies the main Use Cases of the envisioned IMS LD Authoring Tool and compares the existing IMS LD implementations based on how well they address the identified Use Cases. From this comparison, it is evident that, although there are a number of existing IMS LD Authoring Tools (such as Reload LD Editor, CopperAuthor, ASK-LDT), in TENCompetence a higher-level graphical Learning Flow tool is needed to enable practitioners visualize and assemble Units of Learning easily, that will support learning designers and practitioners through the entire lifecycle of the authoring process. The design paradigm of this tool should be closer to common practices on designing pedagogical scenarios, rather than to the XML-based structure of the IMS Learning Design specification. Thus, in this document the architectural design of the TENCompetence IMS LD Authoring Tool, towards addressing the authoring needs for learning designers and educational practitioners expressed in the Use Cases, is presented.

Future work includes (a) the definition of an algorithm for machine (automatic) transformation of workflow design paradigms to the lower XML language of IMS LD and vice versa, (b) the initial implementation of the TENCompetence IMS LD Authoring Tool prototype, (c) the elaboration of a set of examples to be used for testing the TENCompetence IMS LD Authoring Tool, and (d) the planning and execution of a usability evaluation study.

---

1 This plan was revised following changes in the consortium see main body of this report)
6 ReCourse V.1.0, the TENCompetence Learning Design Authoring tool

In month 14, when it became clear that it would not be possible to develop the planned inference engine, a strategy was required to repair the lack of an effective LD authoring environment. This has been done by returning to the core task defined in the TENCompetence Detailed Implementation Plan 2 (DIP 2) which is defined as

“Extend and evaluate existing open source IMS LD Authoring and Run-time tools to support the requirements of lifelong competence development and to function as components and/or services into the overall TENCompetence system and evaluate the usability of the produced software components/tools.”

The Reload LD Editor is widely accepted as the reference implementation of an IMS LD authoring tool, and it has been widely used in the development of UOLs in projects such as UNFOLD and LearningNetworks, Collage, and in the Design for Learning programme of JISC. Consequently it was a natural choice to focus on this tool particularly as the consortium includes people who have previously worked on Reload.

Creating an effective editing component which has been the principal focus for this task since month 14. This has not, however, been a trivial task.

Firstly, despite its success, the Reload LD Editor has a number of shortcomings for the purposes of the TENCompetence project. While it enables users to access all parts of the specification, it does so using a tree interface which requires users to have a detailed knowledge of the specification. The interface makes use of a traditional GUI toolset of tables, trees, lists and text fields which is available in Eclipse. This set of interface devices leads to a formalised workflow that reflects the hierarchical structure of the XML binding of the IMS specification rather than a natural workflow reflecting processes of pedagogical design or lesson plan design. This is shown in figure 1 below.

This kind of interface may be satisfactory for technical experts, but the target user group for WP6 is teachers and learning designers. We expect them to have an interest in modelling, and to be willing to engage with complex designs, but we aspire to help them as much as possible in working with the metaphors of IMS LD, and do not expect them to have technical skills or understanding.

A much richer GUI framework can be provided by the Eclipse Graphical Editing Framework (GEF). According to the GEF project website:

The Graphical Editing Framework (GEF) allows developers to create a rich graphical editor from an existing application model. GEF consists of 2 plug-ins. The org.eclipse.draw2d plug-in provides a layout and rendering toolkit for displaying graphics. The developer can then take advantage of the many common operations provided in GEF and/or extend them for the specific domain. GEF employs an MVC (model-view-controller) architecture which enables simple changes to be applied to the model from the view.

In the creation of a graphical interface for Reload these features are highly desirable. Unfortunately, however, the GEF is widely recognised to be poorly documented and to have a steep learning curve. Thus a considerable investment of developer time was required to understand how the GEF could be used in the creation of an IMS LD editor. The outcomes were very positive, and have resulted in version 1.0 of the application, which utilizes 2-dimensional drawing objects made available in the GEF, such as those found in flowcharts, UML design tools and business process tools. This rich toolset means that we can now
implement GUIs which are far more intuitive and visual than before (see Figure 4 TENCompetence ReCourse v1.0 graphical interface, developed using the GEF framework). Moreover this can now be done much more rapidly than in the current version of Reload, and so that rapid iterations of interface design and testing with fully functional applications will now be practicable. Figures 1 & 2 on the following page show screen shots of the original Reload application and the new ReCourse application developed by TENCompetence, and give some indication of the possibilities of the new application.

In order to achieve these benefits, a new code base was required. Fortunately not all the code from the current version of Reload needs to be thrown away. Obviously, new GUI panels require new development time, however this process is eased by the existing work and by the GEF framework itself. Some important code can be re-used from the old LD editor, such as:

- Packaging into a Content Package
- General application framework
- Some editing panels
- XML parsing logic (JDOM)

Thus some parts of the code from the old Reload LD Editor could be reused, but in order to implement the new paradigm, it was necessary to develop a new framework based upon GEF. This is a straightforward and clear strategy due to the fact that

- The existing Reload Editor was developed using Eclipse
- A GEF interface for an LD Editor has already been developed

Thus the technical challenges presented by the creation of a graphical interface for the Reload LD Editor have now been overcome. The creation of an effective interface for the creation of Units of Learning has, however, been a major stumbling block in the past, with teachers finding it very hard to agree on an appropriate interface or framework of pedagogic components, or to specify how it should be built\(^2\).

In order to meet this challenge the LD Author development team has entered into a strategic alliance with a user group who are willing to work closely with the development team in developing an appropriate interface. This will enable us to build to the practical requirements of the teachers, rather than asking them to provide an abstract analysis of their needs (or, worse, trying to imagine what they might be).

A strategic alliance with an IMS LD user group

The project team has succeeded in establishing a close working relationship with the IMS LD led by Mark Baxendale at Liverpool Hope University, an Associate Partner of the TENCompetence project. This team is one of the very few groups worldwide who are using IMS LD in producing and running courses with students in an authentic setting. They have
also coordinated a number of past and present projects with teachers within and beyond Liverpool Hope, aimed at improving the effectiveness of IMS LD tools. They are therefore in an excellent position to provide the development team with the input they need for designs. The team in Liverpool Hope wrote an evaluation report on the Reload LD Editor, carried as part of the JISC funded LD4P project. In this report they identified interface issues which have been picked up in the development of ReCourse. Liverpool Hope also contributed to the present development effort some initial mock-ups showing their design ideas generated using GUI Design Studio and Toolbook. This collaboration was particularly timely in helping to make fast progress in the wake of changes in work package following M 6.1,
7 Design requirements for the WP6 LD Editor

Version 1.1, 15 November 2007

Main author Phillip Beauvoir

Contributions:
Dai Griffiths, UB
Paul Sharples, UB
Yongwu Miao, OUNL
Mark Barrett-Baxendale, Liverpool Hope University

Abstract
This document sets out the design requirements which informed the development of version 1 of the TENCompetence Learning Design Authoring Tool, which is named ReCourse.

The document provides a detailed view of the design approach taken by the development team. It is not, however, a reliable guide to the functionality of the tool because

a) some of the functionality in the tool is not described here

b) some of the design proposals in this document were adapted to respond to the ongoing logic of the development process.

The document should be read in conjunction with the Quick Start Guide, available online at http://www.tencompetence.org/ldauthor/, which provides a comprehensive overview of the functionality of the application, and screen shots of the application itself.
7.1 Introduction

The intent of the LD Authoring tool is to support the creation IMS Learning Designs (UOLs) in an easy and intuitive manner. It is based on the Reload LD Editor with new design ideas and drawing on Bill Olivier’s PowerPoint document, “From EML to Learning Design & Planned OS Implementations focusing on RELOAD” and from contributions from Bolton University, Liverpool Hope University and the OUNL. The main theme is simplicity and a task-focused UI that reduces information overload. Whereas the intent of the original Reload LD tool was to expose all of the many LD elements in trees and tables and thereby expose the full extent of the IMS LD specification, the intent of the new tool is to expose only the necessary elements at a structural level broadening the focus and to provide a meaningful perspective on the designer’s intentions.

The TENCompetence Learning Design Authoring tool described here has been named ReCourse. This name indicates the link with the old RELOAD application, but also indicates that this is a substantially different tool.

Input

The LD tool should be able to open IMS compliant LD packages, parse them and present them to the user for editing in the UI. Once opened they can be edited and re-saved. The tool should also be able to open LDs from other users of the tool. The IMS XML format should be preserved at all times.

Output

The first version of the LD Authoring tool will allow the creation of full IMS Learning Designs at Level A. It is hoped that a later version will permit the saving of parts (“chunks”) of IMS Learning Designs (UOLs) and “templates” (see below). A full LD conforms to the IMS LD spec and XML binding, with its imsmanifest.xml file and content packaged in an IMS Content Package (a “Unit of Learning”). A part of a LD consists of a “chunk” of the IMS LD spec, for example an Activity Sequence, a set of Resources. These chunks may be re-used in the editor between different UOLs and also between users using the same tool. The chunks will be saved locally and exported and will appear in selection dialog boxes and toolbars for insertion into UOLs.

Additional information for the LD, such as UI information (colours, positions of graphical elements, etc) descriptions, notes and narratives is saved in the XML file. This will only have meaning for the recipient if they re-edit the UOL in their copy of the LD Authoring Tool.

Templates for UOLs

A feature of the LD tool is known as “templates”. Ordinarily, a UOL contains concrete resources in many places – HTML files for descriptions, resources, feedback descriptions, etc. Some of these are mandatory for a compliant IMS LD (an Activity must have an Activity Description, for example).
It may be the case that the designer may wish to save the structure of the UOL without at this stage providing concrete instances of these resources. In this case the skeletal form of the UOL will be saved with empty place-holders for the missing resources. The UOL can be saved as a template which can be re-used as the basis of a new instance of a UOL. When building a new UOL based on a template the designer will be prompted to provide the necessary resources for completion of the UOL.

In LD Tool v.1 this functionality is provided through an “open as” command, creating a new clone of an existing UOL. The items to be changed can be flagged using the “notes” feature.

Target users and design goals

Target users

The LD Authoring tool is intended for users who have more than a passing interest in pedagogic design, and who are willing to engage with the complexities of planning pedagogic activities with a modelling language. They are not expected to have any programming skills or knowledge of XML.

Those users who do not wish to engage with pedagogic modelling will use the higher level Pre-authoring tool, currently under development by WP6.

The design goals for the previous Reload LD editor tool were:

- Produce an editor fully compliant with IMS LD levels A, B and C
- Implement Content Packaging for the LD
- Present to the user all of the IMS LD elements as per the XML binding, with nothing hidden

The design goals for the new LD Authoring tool are:

- Less “specialist” focus
- Simple to use
- Some LD elements hidden yet available if needed
- Written text and narrative spaces to explain the designer’s intent
- In-situ creation of elements from all windows
- Meaningful language and terms (configurable)
- Different levels of complexity of editing
- Re-use of UOLs and UOL components
- Hide the XML representation
- Activity flow diagram
- Properties and Conditions presented graphically
7.2 Placement of the LD Editor

In terms of closeness to the specification, the ReCourse tool is placed somewhere between a higher-level specific-case tool and the Reload tool.

7.3 The Organiser View

The Organiser View is a tree containing all the user’s Learning Designs, “chunks” of LDs and Resources – UOLS, Activities, Environments etc. These can be labelled and tagged as required.

When a user creates a new Learning Design, a default built-in LD is loaded with useful default elements and descriptive notes. There should be a preference to allow the default LD to be specified as one of:

- Blank
- Built-in
Custom

Another option could be “New LD based on ...” which is basically copying an existing LD and using that one.

The components that can be placed in the Organiser are:

- Whole LDs
- Parts of LD – Activities, Environments, etc
- Resources of files and URLs (basically bookmarks)

An LD or part of LD could be denotes as a “template” so that when opened a new instance of it is created. The template could of course be edited.

The key to the Organiser View is the full use of Copy & Paste and Drag & Drop between it and the rest of the application. Components should be freely available from the Organiser and draggable to and from views.

**Technical notes:**

As the Organiser View has to be persisted regularly it has to have a lightweight backing model. This means that each backing model tree entry should only contain an entry name and a file reference to the entry’s object stored as a file not the object itself.

### 7.4 The LD Editor Window

The Main LD Editor window consists of the standard Eclipse RCP (Rich Client Platform) window with its client Views, Editors and Perspectives.

- **View** – a resizable and hideable window that displays information about a selected item, for example an outline view, an element’s properties, a file system tree.
- **Editor** – a window that reflects the lifecycle of an editable object – open, edit, save, cancel, undo/redo.
- **Perspective** – an arrangement of Editors and Views (aka a “layout”)
- Each of the main IMS LD categories will be edited on a page (tab) of a Multi-Page Editor:
  - Overview
  - Plays, Acts, and Role-Parts
  - Activities
  - Environments
  - Roles

The user should be able to start designing the UOL from any of these pages, but the order of the tabs suggests a sequence of the following:

1. Describe the intent of the UOL, its pre-requisites and learning objectives in the Overview page (we can also put the information relating to the LD “Method” here).
2. Design the overall flow of the UOL in terms of its Plays, Acts and Role-Parts.
3. Consider the Activities involved in the UOL, which Activities should be sequenced, which should be classed as “Learning” and which as “Support”.

4. Consider the supporting Environments used for Activities and Role-Parts. Add the content of each Environment (sub-Environments, Learning Objects, Services).

5. Define useful names and properties for each Role in the UOL

This order is of course up to the designer, and the tab order could be a configurable user preference.

For each editor, only the relevant information pertaining to that editing task should be displayed. Additional information should be displayed in contextual views or available via dialog boxes. These contextual views may be re-sized, positioned or even hidden. This allows the designer to focus on the task in hand.

**LD Objects and References to Objects**

IMS LD objects are stored internally as a referenceable set of Roles, Activities, Activity Structures, Environments, Properties, etc. These are re-used by referencing them (like Shortcuts). Thus, an Activity Sequence is just a series of references or pointers to Activities. In the Reload LD Editor, the user had to firstly create the original object and then reference it later in the LD. In the new LD Authoring tool we should let the user create the object “the other way around”. For example, the user should be able to directly create a Role in a Role-Part in the Sequence Editor; what actually happens behind the scenes is that the newly created Role is added to the Roles “bank” (and the Roles Editor) whilst adding the reference to the Role in the Role Part. As far as the user is concerned they “just created a Role”. Of course, since the newly created Role has been added to the “bank” of Roles, it automatically becomes available for re-use throughout the LD Editor and available for additional editing in the Roles Editor. The “bank” does not exist as a single visible entity in itself. The “Role bank” is the set of Roles in the Role Editor.

**7.5 Overview Editor Page**

This Editor needs to display the following detailing the UOL:
- Title
- URI
- Version
- Level
- Tags
- Description
- Prerequisites
- Learning Objectives
- Fields from the “Method” part of LD (these relate to how and when the UOL is completed)

**7.6 “Course” Editor Page**

**Introduction**

The “Course” Editor is a sub-Editor page of the main LD Editor displaying the Plays, Acts and Role Parts of the LD. It relates to the “method” of a LD.
Requirements

The Course Editor should allow the user to easily visualise the sequence of Acts in any given Play, and also easily see the Role Parts in each act together with the Roles and Activities/Activity Sequences/Environments that they reference.

It should be possible to create anew the following elements from the Course Editor:

- Plays
- Acts
- Role Parts

The user should as well be able to select and reference the following existing elements:

- Roles
- Activities
- Activity Sequences
- Environments

User Interface

The User Interface (UI) for the Course Editor is placed on a tab of the Multi-page LD editor. It is GEF based. It is based on the ideas in Bill Olivier’s EML/LD PowerPoint presentation, but somewhat modified. Double-clicking on any object in the Course Editor opens up a Properties editor for that object. Single-clicking updates additional contextual views.

The main elements that a user needs to see in this Editor are:

- Plays
- Acts
- Role-Parts and their elements

The key to success is to provide an overview of these elements and the relationship between them. It is also essential that the Role-Part sub-elements are shown (Role and Activity), though not the Environments relating to an Activity which would make the UI too cluttered. A list of the Environments that relate to an Activity can be displayed in a separate contextual view, or seen by double-clicking on an Activity to open up a new window.
Figure: Conceptual wireframe diagram for the user interface of ReCourse, showing definition of role parts.
7.7 Activities Editor Page

In IMS LD Level A there are two types of Activity Structure – “Selection” and “Sequence”. Therefore, you can represent these two types in one of two ways - a group of Activities (Selection) or an arrowed flow diagram (Sequence):

Figure - IMS LD Activity Structure types

The following type of conditional branching is not supported in IMS LD Level A but is supported at Levels B and C using Properties and Conditions:

Figure - Conditional Activity branching

Activities cannot be conditionally branched when they are included in an Activity Structure. An Activity Structure can only be set as a sequence or a choice (selection).
At IMS LD Level B, one can specify Conditions and Properties that hide or show Activities in an Act. An Activity can be completed depending on a Property value being set at runtime, and can also trigger the setting of a Property value on completion of the Activity. Conditions relating to these properties are checked at runtime. For example, with 3 Activities, depending on the completion of Activity 1, you could set either Activity 2 or Activity 3 to be visible next. This will need to be presented clearly in the UI.

For the UI we can take a cue from a section of the IMS LD spec doc, "IMS Learning Design Best Practice and Implementation Guide", section 3 (but mainly Section 3.2.4 onwards). This gives the fuel valve example, which provides us with the clues for a UI. The design proposed is related to an Activity diagram tool that lets the user sequence the flow and add branches (e.g. the diagram below). In future releases the nodes could be editable as IMS LD property/conditions, making the node a flow decision point and discoverable within the conditions which control the flow path choice.
Activity Flow taken from IMS LD Best Practice document
7.8 Environments Editor Page

The Environments’ Editor consists of the set of Environments in the Unit of Learning plus each Environment’s components (Learning Objects, Services and references to other Environments). These can be visually grouped in any way that the user wishes, and helpful Note elements added.

A connection between Environments is possible, as represented in the following graphic:

![Environment 1](image1)

![Environment 2](image2)

7.9 Role Editor Page

A Role Editor is simpler to implement, since we can design a hierarchical box set much like a tree, and include some helpful visual aspects:

![Role Editor](image3)

Figure - Role Editor
7.10 Code Editor Page

A possible additional editor/view is a Code View/Editor page, where the resulting IMS XML for the LD is generated “on the fly” and can be viewed and possibly edited. Some considerations:

• To allow the code to be editable would involve issues of “round-trip engineering”, i.e. a change in the code automatically creates a change in the UI “on the fly”, and vice-versa. This is not a trivial exercise!
• If the code is not editable then what use is it to the user?
• If the IMS XML code is not generated until the user performs an explicit “Save As IMS LD”, then the code will not be available at design time unless this action is performed in order to generate the XML code.

7.11 Files and Resources Organiser View

This should display available and used files and resources for the LD. The user should be able to do the following:

• See the used files and Resources in the LD (occurrences)
• Preview the files (simple text and HTML)
• Edit the files (in internal and external editors)
• Organise the files
• Drag and drop files from the View to the Editors
• Drag and drop from the desktop to the View and to the Editors (the latter placing a reference to the file in the View)

7.12 LD Preview Window

An optional window would be LD Preview, where the LD can be “played”. This needs to be However, the functionality for this is not clear:

• In what sense is it a “Preview”? Should it set up dummy users?
• Does it “play” in the sense of being a running Unit of Learning, or does it only provide an indication of how it will appear?
• Does it need to run on a server and CopperCore?

As integration with the SLeD runtime player is planned for early 2008, this feature may not be necessary.
7.13 Item Editor/Viewer

Everywhere in IMS LD is the “ItemModelType”. This is a construct taken from IMS Content Packaging where resource content items are arranged hierarchically like a table of contents. These items can be set to “visible” or “invisible” at run-time by Conditions (for example, when Activity A1 is completed make the following Items visible):

```
<table>
<thead>
<tr>
<th>Item 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item 2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Item 2a</td>
</tr>
<tr>
<td>Item 2b</td>
</tr>
<tr>
<td>Item 3</td>
</tr>
</tbody>
</table>
```

Wherever there is descriptive text, for example in “Learning Objectives” or “Prerequisites”, we see this construct. In the Reload Editor we presented it in a tree-table like this:

```
Learning Objectives
Title: Hello World

Resource Items:

<table>
<thead>
<tr>
<th>File/URL</th>
<th>Title</th>
<th>Visible</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>step1.html</td>
<td>Chapter 1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>step2.html</td>
<td>Chapter 2</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>step3.html</td>
<td>Sub chapter</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>step9.html</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
```

Double-clicking an item in the table previewed the item’s File/URL in a separate Browser view.

In the new LD Author tool we need to improve this, so as to show the user the content, not the tree. We propose 2 modes – “preview” and “edit”. The default mode is “preview” as shown here:
The leftmost button is a toggle between “edit” and “preview” mode. The arrow buttons to the right preview the items in (flat) sequential order. “Edit” mode displays the tree-table as before:

The items can be added, deleted and moved using full drag and drop functionality.
7.14 Code architecture

The LD design is to be saved in the conformant IMS LD XML format. Additional UI data is saved in an extension tag in the <metadata> tag of the <learning-design> tag. There is an XSD schema file to validate this.

7.15 File formats

The LD design is to be saved in the conformant IMS LD XML format. Additional UI data is saved in an extension tag in the <metadata> tag of the <learning-design> tag. There is an XSD schema file to validate this.

7.16 Serialisation

All XML serialisation is taken care of by the JDOM library (http://www.jdom.org).

7.17 Database

At the time of writing, data objects are saved as separate files. In a later version, it would be advisable to store the files in a local database file for convenience and searching.

7.18 Technical considerations

7.18.1 Undo/Redo Frameworks

As GEF has its own Undo/Redo Command stack it is best, when using an Editor that extends or contains a GraphicalEditor class, to create for all actions GEF Commands and execute them on the Editor's CommandStack. However, if the Editor or View is not GEF based then we should use the standard Eclipse AbstractOperaton and OperationHistory Undo/Redo framework.

7.18.2 GEF

How is an object added to the canvas when you drag one from the palette?
1. ScrollingGraphicalViewer is a drag drop listener. It registers the drop action and requests a Command from the FreeFormDiagramPart (AbstractEditPart) which is the main EditPart for the GraphicalViewer and has a parent Data Model object.

2. FreeFormDiagramPart has a FreeFormDiagramLayoutEditPolicy for its layout behaviour. The requested Command comes from here - getCreateCommand(CreateRequest request). This returns a new CreateGraphicalObjectCommand. This in turn asks the DataElementFactory for a new object instance based on the Class type.

3. The Command is executed by the Drop handler. The CreateGraphicalObjectCommand (CreateObjectCommand) adds the newly created object instance to the Data Model parent object. This in turn fires a Property Change event (PROPERTY_CHILDREN) to registered listeners.

4. FreeFormDiagramPart is a registered Property Change listener. On this event it calls refreshChildren(). This examines the child Edit Parts and sees that the backing Model has a new member but no corresponding EditPart. One is created and this in turn calls addChildVisual(EditPart childEditPart, int index) which creates the corresponding Figure to be added to the ContentPane (the parent Figure).

5. The object is then selected (called from TemplateTransferDropTargetListener).
8 Evaluation of ReCourse, TENCompetence Authoring Component

Compiled by Mark Baxendale, Bill Norton and Dai Griffiths, December 2007

8.1 Introduction

This report, carried out in the context of the DesignShare project funded by JISC evaluates the ReCourse LD Editor V.1. This application has been developed in the context of the TENCompetence project funded by the European Commission, and builds on the Reload LD Editor (although it is a largely new development project, with a different code base). At an early stage in development the DesignShare team at Liverpool Hope University provided input into user needs and design through the evaluation of the Reload LD Editor and subsequent user interface proposals. The present evaluation builds on this work.

The user group for the ReCourse Learning Design Editor is composed of learning designers, whose role is to define flows of pedagogic activities to be carried out by cohorts of learners and teachers.

The output of ReCourse is an interoperable lesson plan, which can be run by a learning design player which takes responsibility for coordinating learning activities and making available appropriate resources and services.

To achieve this it is necessary to use a formal notation to model pedagogic activities by representing the roles, activities, resources, workflows etc which are used in the educational process. The notation used is IMS LD, which aims to constrain as little as possible the structures which the learning designer can represent. Because of this IMS LD modelling language is extensive and has a certain irreducible complexity which can only be off set by constraining the options available to the learning designer. Additional barriers are the expression of the language in XML (which is hard for non experts to understand or edit), and user interface problems with tools which are made available to authors. The ambition of the ReCourse editor is to provide a tool which makes available the whole of the IMS LD modelling language in as clear a way as possible, and which minimises the barriers created by either the use of XML or shortcomings in user interfaces.

Consequently the user group “learning designers” can be further qualified as being people who:
1. Are interested in modelling pedagogic activities to be carried out by others (or by themselves at a later date)
2. Have a basic understanding of how to use a computer, but no technical knowledge of editing XML
3. Have an understanding of the concepts and metaphors used in IMS Learning Design

These people may be pedagogic experts whose professional role is to design courses. Alternatively they may be teachers who are sufficiently interested in pedagogic modelling and/or in the provision of technical support for learning to have taken the trouble to understand the basic concepts and metaphors of IMS Learning Design.

This presents a “bootstrapping” difficulty. There are no present tools which are suited for such users, and consequently there is no appropriate existing user group. Indeed it is the purpose of the ReCourse editor to make it possible to create such a group of users. Consequently we have worked with teachers who are well disposed to pedagogic modelling, but who do not yet have an understanding of the concept and metaphors of IMS Learning Design.
Accordingly the objectives of the present evaluation are to establish if the respondents
1. experience ReCourse as being easy to use.
2. are able to engage successfully with IMS LD with the guidance of the worksheet.
3. find ReCourse easier to use than Reload LD Editor

The longer term goal of the work carried out, however, is to carry out research on IMS Learning Design itself, examining its effectiveness as a modelling language, the way in which its metaphors can be represented, adapted and presented for different user groups, and the impact which it has on educational practice. Once the development team is satisfied that ReCourse is fit for purpose (through this evaluation and other feedback from users), then it can be used in more extended studies, together with other parts of the infrastructure required for authoring and running IMS LD based learning activities. For this purpose it will be important to provide users with training in the concepts and metaphors of IMS LD prior to their use of the infrastructure.
8.2 Methodology

Target user group
As discussed more extensively in the introduction, the target user group for this evaluation is teachers who are willing to engage in understanding the metaphors and structures of the IMS LD specification.

Respondents
The ReCourse editor was used by nine respondents, who were shown a UoL running SLed and given a very brief introduction to learning design.
These consisted of
- Four lecturers at Liverpool Hope University (LHU)
  - Two Computing lecturers, both of whom had previously taken part in a Reload editor workshop (several months had passed since).
  - A Sport Science and a Psychology lecturer who had no previous experience of IMS LD or Reload
- Five further education lecturers at St Helens College (SHC)

Worksheet and activities carried out
Participants attended a single session during which they were given a worksheet to follow (appendix 1). The worksheet involved producing a simple unit of learning:
- Adding details
- Inserting a title
- Creating activities
- Create six activities and rename
- Creating Activity Structures
- Create a sequence and a selection
- Adding activity descriptions
- Import files and assign to activities
- Setting up the course
- Add a phase
- Associate role parts with activity sequences
- Adding Environments
- Import files
- Create an environment and learning object, associate with a file and link to an activity

All participants completed a questionnaire at the end of the session (appendix 2).
In order to maximise comparability with the results of the Reload LD Editor evaluation, this questionnaire, and the activities carried out by participants in the evaluation, were modelled closely on those used in the evaluation of the Reload LD Editor carried out earlier in 2007.
The results of the questionnaires were compiled, and a summary table created showing the mean and the standard deviation.
In analysis of the questionnaires scores for questions were adjusted so that a higher score always indicates a higher degree of satisfaction. For example a mean of 1.22 for "I have not found this software useful" was inverted to indicated a mean of 3.78, which should be
interpreted as a positive indication of the usefulness of the software. This adjustment makes it much easier to compare the results, without the reader having to remember if a higher or lower score is a better outcome for a particular question.

The results of this evaluation were compared with those for the Reload LD Editor to identify those areas where the ReCourse editor provides an improved environment for learning designers, and those areas where more work may need to be done.

The questionnaire also included an free text questions where participants were invited to comment on their experience of using ReCourse, the interface, how the software facilitates workflow, and ease of use. The main points from these comments are presented in tabular form.

In some cases (with agreement of the participants), screen activity and audio was captured. This material provides a valuable resource which the developers will be able to review in order to analyse in detail the usability issues identified in this report.

An additional interview was carried out with an expert user from the user group, who was involved in both developing design suggestions and in the evaluation of the tool.
8.3 Questionnaire analysis

In Table 1, mean scores for the statements (all but one over 3.4) show that participants were generally very positive about nearly all aspects of ReCourse. Note that the scores have been adjusted so that a higher number indicates higher satisfaction in all cases, even when the question is negative.

High ratings (those with a mean of 4 or over) were given for statements on screen layout, enjoyment in using the software, and whether the different sections of the software made sense. Most importantly perhaps, statements on designing learning activities, successfully creating a Learning Unit and feeling that they could use it in the future, by themselves or with support, were also given ratings over 4. The only statement for which the rating was low (mean = 2.2) was ‘Sometimes I don’t know what to do’. 
Table 1: Mean scores for responses to statements on questionnaires completed in December 2007 (N=9)

<table>
<thead>
<tr>
<th>Statements</th>
<th>Min</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>I find the layout of the screen confusing</td>
<td>3</td>
<td>5</td>
<td>4.00</td>
<td>.707</td>
</tr>
<tr>
<td>The software helps me to design learning activities</td>
<td>3</td>
<td>4</td>
<td>3.89</td>
<td>.333</td>
</tr>
<tr>
<td>I can easily carry out the tasks that I need to</td>
<td>2</td>
<td>5</td>
<td>3.67</td>
<td>.866</td>
</tr>
<tr>
<td>It is difficult to find the functions that I want</td>
<td>3</td>
<td>5</td>
<td>3.67</td>
<td>.707</td>
</tr>
<tr>
<td>I find the terminology used by the software confusing</td>
<td>2</td>
<td>5</td>
<td>3.67</td>
<td>.866</td>
</tr>
<tr>
<td>The different sections of the software make sense</td>
<td>3</td>
<td>5</td>
<td>4.00</td>
<td>.500</td>
</tr>
<tr>
<td>Sometimes I don't know what I should do</td>
<td>1</td>
<td>4</td>
<td>2.22</td>
<td>.833</td>
</tr>
<tr>
<td>It is easy to learn how to use this software</td>
<td>3</td>
<td>5</td>
<td>3.78</td>
<td>.667</td>
</tr>
<tr>
<td>It is easy to navigate around the software</td>
<td>2</td>
<td>5</td>
<td>3.44</td>
<td>1.014</td>
</tr>
<tr>
<td>I have not found the software useful</td>
<td>2</td>
<td>5</td>
<td>3.78</td>
<td>.972</td>
</tr>
<tr>
<td>The software effectively supports my workflow</td>
<td>3</td>
<td>4</td>
<td>3.44</td>
<td>.527</td>
</tr>
<tr>
<td>The software is easy to use</td>
<td>2</td>
<td>5</td>
<td>3.67</td>
<td>.866</td>
</tr>
<tr>
<td>I enjoy using this software</td>
<td>3</td>
<td>5</td>
<td>4.00</td>
<td>.500</td>
</tr>
<tr>
<td>The navigation is confusing</td>
<td>2</td>
<td>4</td>
<td>3.56</td>
<td>.726</td>
</tr>
<tr>
<td>The software presents activities in a clear way</td>
<td>3</td>
<td>4</td>
<td>3.56</td>
<td>.527</td>
</tr>
<tr>
<td>I can successfully create a Unit of Learning</td>
<td>2</td>
<td>5</td>
<td>4.00</td>
<td>.866</td>
</tr>
<tr>
<td>I can imagine that I could use the software in future by myself</td>
<td>4</td>
<td>5</td>
<td>4.33</td>
<td>.500</td>
</tr>
<tr>
<td>I can imagine making use of the software if I had support</td>
<td>3</td>
<td>5</td>
<td>4.22</td>
<td>.667</td>
</tr>
</tbody>
</table>

In Table 1, responses were coded 1 (strongly agree) to 5 (strongly disagree) for negative statements, 5 (strongly agree) to 1 (strongly disagree) for positive statements; a high score indicates satisfaction, a low score indicates dissatisfaction.
As the number of participants is not large, frequency counts of responses made to each statement might be more illustrative of the degree of satisfaction with the software (Table 2). While there are a number of neutral responses (two participants responded neutrally to 9 of the questions), participants show a high degree of unanimity in positive responses to statements, either in agreement or disagreement (except for one particular participant who gave negative responses to some statements).
Table 2: Frequency counts of responses to statements in questionnaire completed in December 2007

<table>
<thead>
<tr>
<th>Statements</th>
<th>Strongly agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strongly disagree</th>
</tr>
</thead>
<tbody>
<tr>
<td>I find the layout of the screen confusing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>The software helps me to design learning activities</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>I can easily carry out the tasks that I need to</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>It is difficult to find the functions that I want</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>I find the terminology used by the software confusing</td>
<td></td>
<td></td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>The different sections of the software make sense</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sometimes I don't know what I should do</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>It is easy to learn how to use this software</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>It is easy to navigate around the software</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>I have not found the software useful</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>The software effectively supports my workflow</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>The software is easy to use</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>I enjoy using this software</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The navigation is confusing</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The software presents activities in a clear way</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>I can successfully create a Unit of Learning</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can imagine that I could use the software in future by myself</td>
<td>3</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can imagine making use of the software if I had support</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
8.4 Free text responses

The questionnaire also included an free text questions where participants were invited to comment on their experience of using ReCourse, the interface, how the software facilitates workflow, and ease of use.

The great majority of the comments are very positive. In those cases where users had previously used Reload they explicitly state that Recourse is much easier to understand and use.

Table 3: Please summarise your experience of using ReCourse

<table>
<thead>
<tr>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basically easy to use - First task should have been to identify elements on screen. Bit of confusion over name of &quot;Environment&quot; in menu &amp; also in Properties.</td>
</tr>
<tr>
<td>For someone with no background in this kind of area, the prog. was relatively simple to follow. I would suggest a few amendments to make the handout a little easier to follow but then others using this will no doubt have more of an instinct for these things?</td>
</tr>
<tr>
<td>I found it an enjoyable experience</td>
</tr>
<tr>
<td>I would simplify some of the terminology in the guide. Keep features as similar as possible to popular applications. I felt very slow this time but now am more familiar with the software and would be quicker/less frustrated next time.</td>
</tr>
<tr>
<td>much more user friendly than the first version used. Good instructions. Would consider using in teaching and learning.</td>
</tr>
<tr>
<td>Quite easy to use, with guidance. This would quickly become intuitive.</td>
</tr>
<tr>
<td>Quite [a bit] harder than test interface but more simple than Reload - once into the instructions it became clearer.</td>
</tr>
<tr>
<td>The experience was intuitive. There was a high level of consistency throughout which helped in other sections.</td>
</tr>
<tr>
<td>Was quite complicated.</td>
</tr>
</tbody>
</table>

Table 4: Please comment on the Interface

<table>
<thead>
<tr>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean and easy to use.</td>
</tr>
<tr>
<td>Clear, easy to use. Displayed in traditional style.</td>
</tr>
<tr>
<td>Good interface; but potentially misleading (2 resource tabs).</td>
</tr>
<tr>
<td>Interface was much simpler than Reload &amp; was structured in a less complex manner.</td>
</tr>
<tr>
<td>Mostly fine, but could tally up a little more closely with the handout.</td>
</tr>
<tr>
<td>Nice to the eye, clear and easy to follow</td>
</tr>
<tr>
<td>Not too complicated. More colour?</td>
</tr>
<tr>
<td>The LD interface was free from clutter and presented a clear interface which was not initially confusing or intrusive.</td>
</tr>
<tr>
<td>Was quite complicated to look at until I had been using it.</td>
</tr>
</tbody>
</table>
**Table 5: Please comment on how the software facilitates workflow**

<table>
<thead>
<tr>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Difficult to answer at the present time.</td>
</tr>
<tr>
<td>Easy to follow</td>
</tr>
<tr>
<td>Good.</td>
</tr>
<tr>
<td>Like drag and drop although I struggled to familiarise myself with some features.</td>
</tr>
<tr>
<td>The software facilitating the flow of work was very well structured and progressive</td>
</tr>
<tr>
<td>The workflow enhanced the facilitation of the learning</td>
</tr>
<tr>
<td>would need to use more to make a considered judgement</td>
</tr>
</tbody>
</table>

Two participants did not respond

**Table 6: Please comment on the software’s ease of use**

<table>
<thead>
<tr>
<th>Comment</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy to use with some guidance - this would quickly become intuitive.</td>
<td>Fairly straightforward.</td>
</tr>
<tr>
<td>Good.</td>
<td>Had a lot of different buttons to use</td>
</tr>
<tr>
<td>Much easier to use than the previous version.</td>
<td>Okay with the workbook - 'help' facility, which talks you through things in similar way.</td>
</tr>
<tr>
<td>Probably much easier the second time around!</td>
<td>The software was much easier than Reload.</td>
</tr>
<tr>
<td>Yes - much better than reload first time round</td>
<td></td>
</tr>
</tbody>
</table>
8.5 Further analysis

Those respondents who gave permission were video recorded as they used the software, and notes were taken. This provides a body of evidence for further analysis of specific usability issues for analysis by the development team. As an example of the kinds of evidence provided, initial study of these materials has highlighted the following issues:

- More than one user experienced difficulties
  - Locating the Palette
  - Finding the Edit button on the activities editor
  - Not confirming an action
- One user experienced difficulties
  - Struggled to find New icon
  - Dragging files into the files pane, resources section
  - Following the section tabs
  - Confused Environments tab with the Environments button on the Activity properties panel
  - Seeing preview
  - Getting an overview of structuring
  - Arranging icons
  - Selecting resources in the edit window
8.6 Expert inspection

The editor was used by an LD4P team member who has considerable practical experience of authoring IMS LD and using the Reload LD editor. It should be noted that this respondent (unlike the others involved in this evaluation) had some input at the design stage of development work, and in some sense be less objective. On the other hand the depth of the evaluation is substantially greater, being the result of more extensive engagement with the tool. The main comments made in expert feedback are as follows:

The ReCourse editor has been used to set up test UoLs in drawing up worksheets for evaluations. It was found to be far easier to use than Reload, especially the activities section—this is very enjoyable to use. Also, observing novice practitioners working with the software, this is the area that they seem to get to grips with very quickly.

The section that seems a little opaque, and seems to cause problems to novices, is the item table. This still feels rather “clunky”, so that to add a resource to an activity one has to follow a few steps: open up the properties panel; add a new item to the table, then choose a resource. Also, unless one has renamed the resource to something meaningful in the resources section, it is not obvious which resource is being selected. Although there is a tool tip, this is obviously not active until the resource has been selected.

The overview section is useful, although as an experienced user the author tended to navigate using the tabs.

The Completion icon leading onto the packaging is confusing, because at first sight this appears to refer to the author completing the UoL.
8.7 Conclusions

From the above data we may conclude that:

1) The respondents experienced ReCourse as being easy to use. There were high mean scores for statements related to interface and use of the software. “Easy to use” and “easy to learn” were also high, but slightly lower. This may be interpreted as that the users The following are on a five point scale, with five indicating a positive result.
   a) interface:
      i) future use of the software unaided (4.3)
      ii) screen layout (4.0)
      iii) enjoyment in using the software (4.0)
      iv) the different sections of the software make sense (4.0)
   b) ease of use
      i) easy to use (3.7)
      ii) easy to learn (3.8)

2) The respondents were able to engage successfully with IMS LD using ReCourse. This is shown by the very high mean scores for success in creating a UOL, and for expectation that in future the software could be used unaided. This does not demonstrate that ReCourse can be used by any teacher unaided in modelling pedagogy. It should be remembered that very clear tasks were given to respondents in the worksheets, and that in normal circumstances users would have the more demanding task of creating or adapting activities which they decided on themselves. Indeed it seems likely that the relatively low score for “knowing what to do” (when taken in combination with the high scores for usability and future use) is a function of respondents unfamiliarity with the IMS LD concepts and metaphors. The results do, however, indicate that ReCourse is suitable for use by non-technical users as the vehicle through which they develop skills in pedagogic modelling with IMS LD.
   i) successfully create a UOL (4.00)
   ii) future use unaided (4.3)
   iii) know what to do (2.2)

3) ReCourse is substantially easier to use than Reload. A comparison of the results for ReCourse compared with those for the same evaluation activities for Reload indicate show improved results for all statements. The greatest improvement is in usability (as might be expected from the strong indicators for ease of use described in (1) above). In the following table we present the improvement for each statement, calculated as (mean score in the ReCourse evaluation) minus (mean score in the Reload evaluation). This finding is supported by the feedback from the expert user.
<table>
<thead>
<tr>
<th>Statement</th>
<th>(ReCourse mean) - (Reload mean)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I find the layout of the screen confusing</td>
<td>1</td>
</tr>
<tr>
<td>The software helps me to design learning activities</td>
<td>0.86</td>
</tr>
<tr>
<td>I can easily carry out the tasks that I need to</td>
<td>0.9</td>
</tr>
<tr>
<td>It is difficult to find the functions that I want</td>
<td>0.84</td>
</tr>
<tr>
<td>I find the terminology used by the software confusing</td>
<td>0.84</td>
</tr>
<tr>
<td>The different sections of the software make sense</td>
<td>0.67</td>
</tr>
<tr>
<td>Sometimes I don't know what I should do</td>
<td>0.28</td>
</tr>
<tr>
<td>It is easy to learn how to use this software</td>
<td>1.23</td>
</tr>
<tr>
<td>It is easy to navigate around the software</td>
<td>0.54</td>
</tr>
<tr>
<td>I have not found the software useful</td>
<td>0.48</td>
</tr>
<tr>
<td>The software effectively supports my workflow</td>
<td>0.41</td>
</tr>
<tr>
<td>The software is easy to use</td>
<td>0.9</td>
</tr>
<tr>
<td>I enjoy using this software</td>
<td>0.87</td>
</tr>
<tr>
<td>The navigation is confusing</td>
<td>0.72</td>
</tr>
<tr>
<td>The software presents activities in a clear way</td>
<td>0.72</td>
</tr>
<tr>
<td>I can successfully create a Unit of Learning</td>
<td>0.67</td>
</tr>
</tbody>
</table>
8.8 Appendix 1 to ReCourse evaluation: Participant worksheet

8.8.1 Introduction

8.8.1.1 Creating a Learning Design: Introduction to Motion

This document takes you through the authoring of a simple learning design. The ReCourse software is a stand-alone desktop application for editing Learning Designs. This document is restricted to the use of ReCourse. A completed Learning Design would be packaged up (called a unit of learning) and published to a Learning Design “playing” system such as SLeD. Below is a screen shot of the completed design running in SLeD.
8.8.1.2 Preliminaries

A folder called “workshop” has been created on the desktop. This holds all the files for your design in ReCourse.

Create a new design:

Click the "New" icon
Click on LD Learning Design
Click "Next".

At this point another wizard opens and you are asked to enter a name for the Learning Design. The name you attribute to the LD will appear in the Organiser and has no meaning outside of the Editor. Click the "Browse..." button and navigate to (and select) the “workshop” folder on the desktop:
Click “Finish”, and you will be taken to the editor with your design open.
8.8.1.3 An Overview of the ReCourse Editor

To navigate to different sections:

8.8.1.4 Adding details

We need to include details for the Learning Design— we will just give it a title. In the overview editor do the following:

Give the LD a name by including it in the **Title** bar
8.8.1.5  Ignore the rest of this screen

8.8.1.6  Creating activities

Our next task is to include the various activities that the design is comprised of. The diagram below shows the activities and how they are structured (grouped) into sequences or selections.

You will be creating the first 6 activities.

Click on the Activities tab. You will be presented with a canvas with some default activities (and an activity structure) already created:
Delete all the default items on the canvas, so that it is blank.
Drag a learning activity from the Palette onto the canvas:
Double-click on the activity’s title and rename it “Introduction: Motion”:

Repeat this process for the other activities. Only include the first six. The canvas should look similar to the one below:

We now need to specify how the student will complete each activity. In this case, we will allow them to indicate themselves that they have completed the activity. Double-click on one of the activities- a property panel will open up.

Click on Completion Rule:
Select User Choice:

Repeat steps 10 and 11 for the other activities.

8.8.1.7 Creating Activity Structures

The next task involves the creation of activity structures which bundles all of the activities together to create an organised composite e.g. in a class a report activity (an activity structure) may consist of the students doing a research task, a literature review task, critical evaluation task, reflection task etc. (the activities).

If we look at the activity diagram, which shows the organisation and flow of a unit of learning, we can see there are three natural groups of activities. You will be creating the first two.

Setting up a sequence: the “Intro to Motion” activity structure

To do this:

In the activities section, drag an activity structure onto the canvas:
Name the activity structure “Intro to Motion”

Drag the three activities Introduction: Motion, Questions on Motion and Answers on Motion into the activity structure:

Ensure the activities are in the correct order. If not, you can drag them up or down.

You need to specify whether the structure is a sequence or selection: double-click the activity structure and in the properties panel set the structure type to *Sequence*.
Now create a second activity structure, “Aspects of Motion”. This is a selection of the other three activities, *Motion with words, Motion with diagrams, Motion: v-t graphs.*

Create a new empty activity structure on the canvas

Name the activity structure “**Aspects of Motion**”

Drag the appropriate activities into the structure and order them

Select the structure type to be selection

The canvas should look similar to the one below:
8.8.1.8 Now save your work.

8.8.1.9 Adding activity descriptions

As a minimum, each activity must be associated with an activity description, which the learner will see when she clicks on the activity as they run through your design. In this case, a set of text documents have already been produced.

The fast way to do this is to drag the resources folder into the ReCourse Files pane in the Resources section:

Click on the Resources tab

Find the Activity Description folder on the desktop and drag it onto the Contents folder in the Files pane:
Each of the files now needs to be associated with a resource so that it will be included within the design.

Delete the default resources:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>File/URL</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA1_Description</td>
<td>content/l01-description.html</td>
<td>webcontent</td>
</tr>
<tr>
<td>SA1_Description</td>
<td>content/SA1-description.html</td>
<td>webcontent</td>
</tr>
<tr>
<td>LO1_Description</td>
<td>content/l01-resource.html</td>
<td>webcontent</td>
</tr>
</tbody>
</table>

Drag *Introduction to motion.txt* onto the resources pane

Repeat for the files Questions about Motion.txt, Answers about Motion.txt, Motion with words.txt, Motion with diag.txt, Velocity time graphs.txt

You now need to associate each activity with a description file:

Go back to the *Activities* section

Double-click on “Introduction: motion”
On the properties pane, click *Description*, then click the *Edit* button

Click the *Add New Item* button

Name the item “Description”

Click the pull-down arrow under the Resource column, and choose the first resource
Repeat steps 13 to 17 for the other activities. The table below associates an activity with an activity description
### Activity Name | Activity Description Doc
---|---
1) Introduction: Motion | Introduction to motion.txt
2) Questions on Motion | Questions about Motion.txt
3) Answers on Motion | Answers about Motion.txt
4) Motion with words | Motion with words.txt
5) Motion with diagrams | Motion with diag.txt
6) Motion: v-t graphs | Velocity time graphs.txt

#### 8.8.1.10 Roles

The roles are the learners, teachers and/or facilitators that will be involved in the unit of learning. When a run of the design is set up, users are assigned to roles. There is a single learner role for this particular design.

The editor includes a single learner role by default whenever a new learning design is created but to check:

Click on the **Roles** tab, at the bottom of the editor window.

In the top left hand panel of the LD window we can see there is a single learner role and a single staff role by default.
Delete the staff role as it is not required

**8.8.1.11 Setting up the course**

The course area is where everything is pulled together in an overall organisation.

If we look at the activity diagram we can see that it has been divided up into three phases (the dashed boxes):

- Introduction Motion
- Aspects of Motion
- Q&A and further info on Motion.

Phases are ways of organising learning within a design and will reflect how you have designed your activities for a class. Whereas activity structures organise activities for individual learners, phases organise them for all users, so that they are a means of synchronising group activity (e.g. a learner cannot progress beyond the first phase until all other users have completed it).

**Note**

Each phase in the worksheet only contains one activity structure. However, a phase can contain *many activity structures* if it suits your needs. Also note that activity structures can contain other activity structures.

To summarise:

- Phase 1 will contain an activity structure that is a sequence.
- Phase 2 will contain an activity structure that is a selection.
- Phase 3 will contain an activity structure that is a sequence.

Click on the *Course* Tab.
The role parts define which role does what activity or activity structure within a phase.

Drag the *Intro to motion* activity structure onto the canvas

Click on the *Connection* tool on the Palette, and join the Learner role part to the activity structure

We now need to define how the phase will complete. We want it to complete when all roles have finished it (this is how we can set group synchronisation points).

Double-click the phase icon, and, in the properties panel, click on *Completion Rule*
Select When the following Role Parts have been completed, and then select the Role Part:
Create a new phase (click the green + icon next to Phases)
Making sure the new phase is selected, drag a learner role onto the canvas
Associate the Learner role part with the “Aspects of motion” activity structure
8.8.1.12 Adding Environments

Environments are used to hold resources (e.g. learning objects, communication tools, websites etc.) that will be used with various activities. Any activity that requires a resource must be linked to an environment.

The next task is to provide environments for the activities. The table below specifies the environments. An environment holds references to the resources and services used by activities.

<table>
<thead>
<tr>
<th>Environments</th>
<th>Associated activities</th>
<th>Associated LOs</th>
<th>Associated Learning Support Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Intro: Motion</td>
<td>Introduction: Motion</td>
<td>Intro - LO</td>
<td>what.html</td>
</tr>
<tr>
<td>2) Motion with words</td>
<td>Motion with words</td>
<td>Motion_words-LO</td>
<td>words.html</td>
</tr>
<tr>
<td>3) Motion with images</td>
<td>Motion with images</td>
<td>Motion_images-LO</td>
<td>motion.html</td>
</tr>
<tr>
<td>4) velocity_time</td>
<td>Motion: v-t</td>
<td>Vel_time-LO</td>
<td>veltime.html</td>
</tr>
</tbody>
</table>

To create an environment:
Click on the Environment tab

Rename the default environment to "Intro: Motion"
We now need to attribute a Learning Object to the environment. To do this we need to import some more files.

Find the LO-Resources folder on the desktop and add it to the files pane in the Resources section
Find What.html and add it as a resource

Go back to the Environments section and double-click the default learning object

In the properties panel click Resources

Click the Edit button

Change the resource to the last resource you added
We now need to associate the learning activity with the environment

Return to the activities section
Double click “Introduction: Motion”
Click on Environments, and select the environment that you renamed

END
8.9 Appendix 2: Questionnaire

Evaluating ReCourse

Experience questionnaire 1

The purpose of this questionnaire is to obtain an indication of your experience of first use of the ReCourse

Please summarise your experience of using ReCourse (please continue over if there is not enough space)

Please briefly comment on:

The interface

How the software facilitates workflow

The software’s ease of use

Name*
<table>
<thead>
<tr>
<th>Statement</th>
<th>Strongly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Agree</th>
<th>Strongly agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>I find the layout of the screen confusing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The software helps me to design learning activities</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can easily carry out the tasks that I need to</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It is difficult to find the functions that I want</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I find the terminology used by the software confusing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The different sections of the software make sense</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sometimes I don’t know what I should do</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It is easy to learn how to use this software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It is easy to navigate around the software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I have not found the software useful</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The software effectively supports my workflow</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The software is easy to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I enjoy using this software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The navigation is confusing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The software presents tasks in a clear way</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can successfully create a Unit of Learning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can imagine that I could use the software in future by myself</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can imagine making use of the software if I had support</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9 The contribution made by the ReCourse LD Editor

Achievements of work with the TENCompetence LD Authoring component may be summarised as follows:

1. Work on this task was initially oriented towards the establishment of a system using an inference engine for converting high level representations of learning designs into IMS LD. The withdrawal of the lead partner and technology provider from the work package led to a change of focus, and a delay in the deliverables corresponding to DIP 1, but deliverables for DIP 2 are on course.

2. A revised technical strategy has been defined, building on the achievements of the Reload LD Editor, the reference implementation of a Learning Design editor (as originally described in the Description of Work). The architecture of the application has been revised, in part to make it possible to use the Eclipse Graphical Editor Framework (GEF), and in part to redress shortcomings in the original Reload implementation.

3. The problem of identifying appropriate interface designs to implement and test has been resolved by establishing a close collaboration with a group of teachers associated with Liverpool Hope University who use IMS LD with learners on a regular basis.

4. Version 1.0 of the ReCourse Learning Design Editor has been developed, released and evaluated. This has the following features
   a. Open, edit and save IMS LD UOLs at level A. This provides the basic functionality equivalent to that used by Reload authors who were not technical experts. This is demonstrated by the fact that evaluation of ReCourse has been able to duplicate the tasks given to users in the evaluation of Reload in trials carried out independently of the project.
   b. Open IMS LD UOLs at levels A, B & C (but not represent the level B & C elements)
   c. Graphical representation of IMS LD elements. On the one hand this makes the interface easier to understand and use (the user can make structures which make sense to them, and use drag and drop) and also has semantic value (e.g. an arrow between a role and an activity automatically creates a role part).
   d. The problem of exchanging graphical information associated with a Unit of Learning (UOL) is addressed by including this information as XML in the manifest of the UOL. Another application which opens the UOL can be written so that it interprets the graphical information which describes the representation of the UOL in the authoring environment. If this is not done then the graphical information will be ignored.
   e. Unlike other available LD Editors, ReCourse uses the IMS LD XML as the file format for the application’s internal working. This removes the import-export process which is required, for example, in Reload. This also ensures compliance with the specification.
   f. A resource manager has been included, resolving a number of shortcomings in the original Reload application.

5. Evaluation of the ReCourse application has shown that
   a. ReCourse was seen to be easy to use
   b. Non-technical users were able to engage successfully with IMS LD
   c. ReCourse is substantially easier to use than Reload IMS LD Editor